

WHAT IS CLAIMED IS:

1. A method for checkpointing processes on a virtual machine executing within a device, the method comprising:

5 executing a process within the virtual machine, wherein the virtual machine comprises a virtual machine virtual memory manager;

 wherein said executing the process comprises:

 the process referencing an object in a virtual heap during execution;

 wherein the virtual heap comprises an in-memory heap and a store heap;

10 wherein, if the referenced object is in the store heap and not in the in-memory heap when referenced by the process, the virtual machine virtual memory manager copying a section of the store heap comprising the referenced object from the store heap to the in-memory heap;

 the process accessing the referenced object in the in-memory heap; and

15 checkpointing a state of the process executing on the virtual machine to a first memory space, wherein said checkpointing comprises the virtual machine virtual memory manager flushing one or more sections of the in-memory heap to the store heap.

2. The method of claim 1, wherein the one or more flushed sections comprise
20 new objects or modified objects in regards to objects stored in the store heap prior to said flushing.

3. The method of claim 1,
 wherein said checkpointing the state of the process executing on the virtual
25 machine to the first memory space comprises storing data describing one or more leases to services for the process, wherein the one or more services are external to the virtual machine on which the process is executing, and wherein the leases are grants of access to the one or more services.

002090-B2028560

4. The method of claim 3,

wherein said checkpointing the state of the process executing on the virtual machine to the first memory space further comprises storing a computation state of the virtual machine to the first memory space, wherein the computation state of the virtual machine comprises information about the execution state of the process on the virtual machine.

5. The method of claim 1, further comprising:

repeating said checkpointing the state of the process so that the first memory space stores a plurality of states for the process, and wherein each of the plurality of states for the process stored in the first memory space is a unique state of the process on the virtual machine.

6. The method of claim 1,

wherein said checkpointing the state of the process executing on the virtual machine to the first memory space comprises storing a computation state of the virtual machine to the first memory space, wherein the computation state of the virtual machine comprises information about the execution state of the process on the virtual machine.

7. The method of claim 1,

wherein the store heap for the process is one of a plurality of store heaps for a plurality of processes on the virtual machine.

8. The method of claim 1,

wherein the checkpointed state of the process is one of a plurality of checkpointed states in the first memory space for a plurality of processes on the virtual machine.

9. The method of claim 1,

wherein the device is a network client device.

10. The method of claim 1,
wherein the first memory space is comprised in a first memory device coupled to
the device.

5

11. The method of claim 10,
wherein the first memory device is coupled to the device via the Internet so that
the virtual machine virtual memory manager writing the one or more sections of the in-
memory heap to the store heap comprised in the first memory space occurs over the
10 Internet.

12. The method of claim 1,
wherein the first memory space is comprised in a non-volatile memory.

15

13. The method of claim 12,
wherein the non-volatile memory is a flash memory;
wherein the store heap comprises a plurality of cache lines; and
wherein each of the sections of the store heap comprise one or more of the
plurality of cache lines.

20

14. The method of claim 1,
wherein the virtual machine is a Java virtual machine, and wherein the process is
a Java application.

25

15. A method for managing processes on a virtual machine executing within a
device, the method comprising:

executing a process within the virtual machine;
checkpointing a state of the process on the virtual machine to a persistent store;

expiring one or more leases to services for the process on the virtual machine;
stopping the process execution on the virtual machine;
reading the stored state of the process from the persistent store;
reconstituting the stored state of the process on the virtual machine;
5 establishing the one or more leases to services for the process on the virtual
machine; and
resuming the process execution on the virtual machine.

16. The method of claim 15, further comprising:
10 stopping execution of the virtual machine within the device consequent to said
stopping the process execution on the virtual machine; and
restarting execution of the virtual machine within the device prior to said reading
the stored state of the process from the persistent store.

15 17. The method of claim 15,
wherein the execution of the virtual machine within the device is not stopped
between said stopping the process execution on the virtual machine and said resuming the
process execution on the virtual machine.

20 18. The method of claim 15,
wherein the state of the process comprises:
a heap for the process, wherein the heap comprises code and data for the
process executing on the virtual machine.

25 19. The method of claim 15,
wherein the state of the process comprises:
data describing the one or more leases to services for the process on the
virtual machine, wherein the data describing the one or more leases is used in said
establishing the one or more leases to services for the process on the virtual machine.

checkpointing the state of the process on the virtual machine to a first persistent heap for the process in the plurality of persistent heaps comprised in the persistent store.

5 25. The method of claim 15,
 wherein the virtual machine comprises a first in-memory heap for caching pages
 for use by the process, wherein the pages comprise code and data for the process;
 wherein the persistent store comprises a virtual heap for storing pages flushed
 from the first in-memory heap; and
10 wherein said checkpointing the state of the process on the virtual machine to the
 persistent store comprises:
 storing one or more pages from the first in-memory heap to the virtual
 heap in the persistent store.

15 26. The method of claim 25,
 wherein said stopping the process execution on the virtual machine comprises:
 deleting the first in-memory heap from the virtual machine.

20 27. The method of claim 25,
 wherein said reading the stored state of the process from the persistent store
 comprises:

 reading the one or more pages from the virtual heap in the persistent store;
 and

25 wherein said reconstituting the state of the process on the virtual machine
 comprises:

 establishing on the virtual machine a second in-memory heap for caching
 pages for use by the process; and

 copying the one or more pages read from the virtual heap to the second in-
 memory heap.

28. The method of claim 15,
wherein the process is a Java process, and wherein the virtual machine is a Java
virtual machine.

5

29. A method for managing processes on a virtual machine executing within a
device, the method comprising:

executing a first process within the virtual machine;

10 checkpointing a state of the first process executing within the virtual machine to a
persistent store;

expiring one or more leases to services for the first process on the virtual machine;

suspending the first process executing within the virtual machine;

15 reading a state of a suspended second process from the persistent store, wherein
the state of the second process was stored to the persistent store prior to said executing
the first process within the virtual machine;

reconstituting the state of the second process on the virtual machine;

establishing one or more leases to services for the second process on the virtual
machine; and

20 resuming the execution of the second process within the virtual machine.

30. The method of claim 29, further comprising:

stopping execution of the virtual machine within the device consequent to said
suspending the first process executing within the virtual machine; and

25 restarting execution of the virtual machine on the device prior to said reading the
state of the second process from the persistent store.

31. The method of claim 29,

wherein the execution of the virtual machine on the device is not stopped between said suspending the first process executing within the virtual machine and said resuming the execution of the second process within the virtual machine.

5 32. The method of claim 29,
 wherein the state of the second process comprises:
 a heap for the second process, wherein the heap comprises code and data
for the second process executing on the virtual machine.

10 33. The method of claim 29,
 wherein the state of the second process comprises:
 data describing the one or more leases to services for the second process
on the virtual machine, wherein the data describing the one or more leases is used in said
establishing the one or more leases to services for the second process on the virtual
15 machine.

 34. The method of claim 29,
 wherein the one or more leases to services include one or more leases to remote
services, wherein the remote services are services provided on devices other than the
20 device within which the process is executing.

 35. The method of claim 29,
 wherein the one or more leases to services include one or more leases to local
services, wherein the local services are services provided on the device within which the
25 process is executing.

 36. The method of claim 29,
 wherein the one or more leases to services include one or more leases to system
services, wherein a system service comprises system code for accessing a resource

external to the process, wherein the system code is provided on the device within which the process is executing.

37. The method of claim 29,

5 wherein the state of the second process comprises:

a stored execution state of the device comprising the virtual machine;

wherein, in said reconstituting the state of the second process on the virtual machine, a current execution state of the device comprising the virtual machine is reconstituted to the stored execution state of the device.

10

38. The method of claim 29,

wherein the persistent store comprises a plurality of persistent heaps for a plurality of processes; and

15 wherein said checkpointing the state of the first process executing within the virtual machine to the persistent store comprises:

checkpointing the state of the first process on the virtual machine to a first persistent heap for the first process in the plurality of heaps comprised in the persistent store; and

20 wherein said reading the state of the second process from the persistent store comprises:

reading the state of the second process from a second persistent heap for the second process in the plurality of heaps comprised in the persistent store.

39. The method of claim 29,

25 wherein the virtual machine comprises a first in-memory heap for caching pages for use by the first process, wherein the pages comprise code and data for the first process;

wherein the persistent store comprises a first virtual heap for storing pages flushed from the first in-memory heap; and

wherein said checkpointing the state of the first process on the virtual machine to the persistent store comprises:

storing one or more pages from the first in-memory heap to the first virtual heap in the persistent store.

5

40. The method of claim 39,

wherein said suspending the first process execution on the virtual machine comprises:

deleting the first in-memory heap from the virtual machine.

10

41. The method of claim 29,

wherein the persistent store comprises a second virtual heap for storing pages flushed from a deleted second in-memory heap for the second process, wherein the pages comprise code and data for the second process;

15 wherein said reading the stored state of the second process from the persistent
store comprises:

reading one or more pages from the second virtual heap in the persistent store; and

wherein said reconstituting the state of the second process on the virtual machine
20 comprises:

reestablishing on the virtual machine the previously deleted second in-memory heap for caching pages for use by the second process; and

copying the one or more pages read from the second virtual heap to the reestablished second in-memory heap.

25

42. The method of claim 29,

wherein the first process and second process are Java processes, and wherein the virtual machine is a Java virtual machine.

[illegible]

43. A system comprising:

a device configured to execute a virtual machine, wherein the virtual machine is configured to execute a first process;

5 a first memory coupled to the device, wherein the first memory is configured to store a store heap for the first process, and wherein the first memory is further configured to store one or more checkpointed states of one or more processes, wherein the store heap is comprised within a virtual heap for the first process;

10 a second memory coupled to the device, wherein the second memory is configured to store an in-memory heap for the first process, and wherein the in-memory heap is comprised within the virtual heap, and wherein the in-memory heap comprises cached portions of the store heap for access by the first process;

15 wherein the device is configured to perform operations on the virtual heap according to a virtual machine virtual heap manager, and wherein the virtual machine virtual heap manager is configured to:

copy a section of the store heap comprising an object from the store heap to the in-memory heap in response to the first process referencing the object in the virtual heap when the referenced object is in the store heap and not in the in-memory heap, wherein the first process accesses the referenced object in the in-memory heap; and

20 checkpoint a state of the first process executing on the virtual machine to the first memory;

wherein, in checkpointing the state of the first process executing on the virtual machine to the first memory, the virtual machine virtual heap manager is further configured to:

25 flush one or more sections of the in-memory heap to the store heap.

44. The system of claim 43,

wherein the one or more flushed sections comprise new objects or modified objects in regards to objects stored in the store heap prior to said flushing.

45. The system of claim 43,
wherein, in checkpointing the state of the first process executing on the virtual
machine to the first memory, the virtual machine virtual heap manager is further
5 configured to:

store data describing one or more leases to services for the first process,
wherein the one or more services are external to the virtual machine on which the first
process is executing, and wherein the leases are grants of access to the one or more
services.

10

46. The system of claim 43,
wherein the virtual machine virtual heap manager is further configured to:

repeat said checkpointing the state of the first process so that the first
memory stores a plurality of states for the first process;

15 wherein each of the plurality of states for the first process stored in the first
memory is a unique state of the first process on the virtual machine.

47. The system of claim 43,

20 wherein, in checkpointing the state of the first process executing on the virtual
machine to the first memory, the virtual machine virtual heap manager is further
configured to:

store a computation state of the virtual machine to the first memory,
wherein the computation state of the virtual machine comprises information about the
execution state of the first process on the virtual machine.

25

48. The system of claim 43,

wherein the store heap stored in the first memory for the first process is one of a
plurality of store heaps stored in the first memory for a plurality of processes on the
virtual machine; and

wherein the checkpointed state of the first process is one of a plurality of checkpointed states in the first memory for a plurality of processes on the virtual machine.

5 49. The system of claim 43,
 wherein the device is a network client device.

 50. The system of claim 43,
 wherein the first memory is coupled to the device via the Internet so that said
10 writing the one or more sections of the in-memory heap to the store heap comprised in the
first memory occurs over the Internet.

 51. The system of claim 43,
 wherein the first memory is a flash memory;
15 wherein the store heap comprises a plurality of cache lines; and
 wherein each of the sections of the store heap comprise one or more of the
plurality of cache lines.

 52. The system of claim 43,
20 wherein the virtual machine is a Java virtual machine, and wherein the first
process is a Java application.

 53. A system comprising:
25 a device configured to execute a virtual machine, wherein the virtual machine is
configured to execute a process;
 a persistent memory device coupled to the device, wherein the persistent memory
device is configured to store a checkpointed state for the process;

wherein the device is further configured to manage the process executing within the device according to a virtual machine process manager, and wherein the virtual machine process manager is configured to:

- store the state of a process executing within the virtual machine to the
- 5 persistent memory device;
- expire one or more leases to services for the process on the virtual machine;
- stop the process execution on the virtual machine;
- read the stored state of the process from the persistent memory device;
- 10 reconstitute the stored state of the process on the virtual machine;
- establish the one or more leases to services for the process on the virtual machine; and
- resume the process execution on the virtual machine.

- 15 54. The system of claim 53,
 wherein the device is further configured to:
- stop execution of the virtual machine within the device consequent to said stopping the process execution on the virtual machine; and
 - restart execution of the virtual machine within the device prior to said
 - 20 reading the stored state of the process from the persistent memory device.

55. The system of claim 53,
 wherein the execution of the virtual machine within the device is not stopped between said stopping the process execution on the virtual machine and said resuming the
- 25 process execution on the virtual machine.

56. The system of claim 53,
 wherein the state of the process comprises:

a heap for the process, wherein the heap comprises code and data for the process executing on the virtual machine.

57. The system of claim 53,

5 wherein the state of the process comprises:

data describing the one or more leases to services for the process on the virtual machine, wherein the data describing the one or more leases is used by the virtual machine process manager in said establishing the one or more leases to services for the process on the virtual machine.

10

58. The system of claim 53,

wherein the one or more leases to services include one or more leases to remote services, wherein the remote services are services provided on devices other than the device within which the process is executing.

15

59. The system of claim 53,

wherein the one or more leases to services include one or more leases to local services, wherein the local services are services provided on the device within which the process is executing.

20

60. The system of claim 53,

wherein the one or more leases to services include one or more leases to system services, wherein a system service comprises system code for accessing a resource external to the process, wherein the system code is provided on the device within which the process is executing.

25

61. The system of claim 53,

wherein the state of the process comprises a stored execution state of the device comprising the virtual machine; and

003090-82028560

65. The system of claim 63,
wherein, in reading the stored state of the process from the persistent memory device, the virtual machine process manager is further configured to:

read the one or more pages from the store heap in the persistent memory
5 device; and

wherein, in reconstituting the state of the process on the virtual machine, the virtual machine process manager is further configured to:

establish on the virtual machine a second in-memory heap for caching pages for use by the process; and

10 copy the one or more pages read from the store heap to the second in-memory heap.

66. The system of claim 53,
wherein the process is a Java process, and wherein the virtual machine is a Java
15 virtual machine.

67. A system comprising:
a device configured to execute a virtual machine, wherein the virtual machine is
20 configured to execute processes;

a persistent memory device coupled to the device, wherein the persistent memory device is configured to store checkpointed states for the processes;

wherein the device is further configured to manage the processes executing within the device according to a virtual machine process manager, and wherein the virtual
25 machine process manager is configured to:

checkpoint a state of a first process executing within the virtual machine to the persistent memory device;

expire one or more leases to services for the first process on the virtual machine;

suspend the first process executing within the virtual machine;

read a state of a suspended second process from the persistent memory device, wherein the state of the second process was stored to the persistent memory device prior to said executing the first process within the virtual machine;

5 reconstitute the state of the second process on the virtual machine;

establish one or more leases to services for the second process on the virtual machine; and

resume the execution of the second process within the virtual machine.

10 68. The system of claim 67,

wherein the device is further configured to:

stop execution of the virtual machine within the device consequent to said suspending the first process executing within the virtual machine; and

15 restart execution of the virtual machine on the device prior to said reading the state of the second process from the persistent memory device.

69. The system of claim 67,

wherein the execution of the virtual machine on the device is not stopped between said suspending the first process executing within the virtual machine and said resuming the execution of the second process within the virtual machine.

20

70. The system of claim 67,

wherein the state of the second process comprises:

a heap for the second process, wherein the heap comprises code and data for the second process executing on the virtual machine.

25

71. The system of claim 67,

wherein the state of the second process comprises:

76. The system of claim 67,

wherein the persistent memory device comprises a plurality of persistent heaps for a plurality of processes; and

5 wherein, in checkpointing the state of the first process executing within the virtual machine to the persistent memory device, the virtual machine process manager is further configured to:

checkpoint the state of the first process on the virtual machine to a first persistent heap for the first process in the plurality of heaps comprised in the persistent memory device; and

10 wherein, in reading the state of the second process from the persistent memory device, the virtual machine process manager is further configured to:

read the state of the second process from a second persistent heap for the second process in the plurality of heaps comprised in the persistent memory device.

15 77. The system of claim 67,

wherein the virtual machine comprises a first in-memory heap for caching pages for use by the first process, wherein the pages comprise code and data for the first process;

20 wherein the persistent memory device comprises a first store heap for storing pages flushed from the first in-memory heap; and

wherein, in checkpointing the state of the first process on the virtual machine to the persistent memory device, the virtual machine process manager is further configured to:

25 store one or more pages from the first in-memory heap to the first store heap in the persistent memory device.

78. The system of claim 77,

wherein, in suspending the first process execution on the virtual machine, the virtual machine process manager is further configured to:

delete the first in-memory heap from the virtual machine.

79. The system of claim 67,

wherein the persistent memory device comprises a second store heap for storing
5 pages flushed from a deleted second in-memory heap for the second process, wherein the
pages comprise code and data for the second process;

wherein, in reading the stored state of the second process from the persistent
memory device, the virtual machine process manager is further configured to:

read one or more pages from the second store heap in the persistent
10 memory device; and

wherein, in reconstituting the state of the second process on the virtual machine,
the virtual machine process manager is further configured to:

reestablish on the virtual machine the previously deleted second in-
memory heap for caching pages for use by the second process; and

15 copy the one or more pages read from the second store heap to the
reestablished second in-memory heap.

80. The system of claim 67,

wherein the first process and second process are Java processes, and wherein the
20 virtual machine is a Java virtual machine.

81. A carrier medium comprising programming instructions executable to
checkpoint processes on a virtual machine executing within a device, wherein the
25 program instructions are executable to implement:

a process executing within the virtual machine referencing an object in a virtual
heap during execution, wherein the virtual heap comprises an in-memory heap and a store
heap;

wherein each of the plurality of states for the process stored in the first memory space is a unique state of the process on the virtual machine.

85. The carrier medium of claim 81,

5 wherein the store heap for the process is one of a plurality of store heaps for a plurality of processes on the virtual machine; and

wherein the checkpointed state of the process is one of a plurality of checkpointed states in the first memory space for a plurality of processes on the virtual machine.

10 86. The carrier medium of claim 81,

wherein the first memory space is comprised in a first memory device coupled to the device.

87. The carrier medium of claim 81,

15 wherein the first memory space is comprised in a flash memory;

wherein the store heap comprises a plurality of cache lines; and

wherein each of the sections of the store heap comprise one or more of the plurality of cache lines.

20 88. The carrier medium of claim 81,

wherein the virtual machine is a Java virtual machine, and wherein the process is a Java application.

25 89. A carrier medium comprising programming instructions executable to manage processes on a virtual machine executing within a device, wherein the program instructions are executable to implement:

checkpointing a state of a process executing within the virtual machine to a persistent store;

- expiring one or more leases to services for the process on the virtual machine;
- stopping the process execution on the virtual machine;
- reading the stored state of the process from the persistent store;
- reconstituting the stored state of the process on the virtual machine;

90. The carrier medium of claim 89,

stopping execution of the virtual machine within the device consequent to said stopping the process execution on the virtual machine; and

91. The carrier medium of claim 89,

92. The carrier medium of claim 89,

a heap for the process, wherein the heap comprises code and data for the process executing on the virtual machine;

wherein, in reconstituting the state of the process on the virtual machine, the program instructions are further executable to implement:

reconstituting a current execution state of the device comprising the virtual machine to the stored execution state of the device.

5

93. The carrier medium of claim 89,

wherein the one or more leases to services include one or more leases to remote services, wherein the remote services are services provided on devices other than the device within which the process is executing.

10

94. The carrier medium of claim 89,

wherein the one or more leases to services include one or more leases to local services, wherein the local services are services provided on the device within which the process is executing.

15

95. The carrier medium of claim 89,

wherein the one or more leases to services include one or more leases to system services, wherein a system service comprises system code for accessing a resource external to the process, wherein the system code is provided on the device within which the process is executing.

20

96. The carrier medium of claim 89,

wherein the persistent store comprises a plurality of persistent heaps for a plurality of processes; and

25

wherein, in checkpointing the state of the process on the virtual machine to the persistent store, the program instructions are further executable to implement:

checkpointing the state of the process on the virtual machine to a first persistent heap for the process in the plurality of persistent heaps comprised in the persistent store.

002090-87028560

97. The carrier medium of claim 89,
wherein the virtual machine comprises a first in-memory heap for caching pages
for use by the process, wherein the pages comprise code and data for the process;
5 wherein the persistent store comprises a virtual heap for storing pages flushed
from the first in-memory heap; and
wherein, in checkpointing the state of the process on the virtual machine to the
persistent store, the program instructions are further executable to implement:
storing one or more pages from the first in-memory heap to the virtual
10 heap in the persistent store.

98. The carrier medium of claim 97,
wherein, in reading the stored state of the process from the persistent store, the
program instructions are further executable to implement:
15 reading the one or more pages from the virtual heap in the persistent store;
and
wherein, in reconstituting the state of the process on the virtual machine, the
program instructions are further executable to implement:
establishing on the virtual machine a second in-memory heap for caching
20 pages for use by the process; and
copying the one or more pages read from the virtual heap to the second in-
memory heap.

99. The carrier medium of claim 89,
25 wherein the process is a Java process, and wherein the virtual machine is a Java
virtual machine.

wherein the persistent store comprises a plurality of persistent heaps for a plurality of processes; and

wherein, in checkpointing the state of the first process on the virtual machine to the persistent store, the program instructions are further executable to implement:

5 checkpointing the state of the first process on the virtual machine to a first persistent heap for the first process in the plurality of heaps comprised in the persistent store; and

wherein, in reading the state of the second process from the persistent store, the program instructions are further executable to implement:

10 reading the state of the second process from a second persistent heap for the second process in the plurality of heaps comprised in the persistent store.

108. The carrier medium of claim 100,

15 wherein the virtual machine comprises a first in-memory heap for caching pages for use by the first process, wherein the pages comprise code and data for the first process;

wherein the persistent store comprises a first virtual heap for storing pages flushed from the first in-memory heap; and

20 wherein, in checkpointing the state of the first process on the virtual machine to the persistent store, the program instructions are further executable to implement:

storing one or more pages from the first in-memory heap to the first virtual heap in the persistent store.

109. The carrier medium of claim 100,

25 wherein the persistent store comprises a second virtual heap for storing pages flushed from a deleted second in-memory heap for the second process, wherein the pages comprise code and data for the second process;

wherein, in reading the stored state of the second process from the persistent store, the program instructions are further executable to implement:

reading one or more pages from the second virtual heap in the persistent store; and

wherein, in reconstituting the state of the second process on the virtual machine, the program instructions are further executable to implement:

5 reestablishing on the virtual machine the previously deleted second in-memory heap for caching pages for use by the second process; and

 copying the one or more pages read from the second virtual heap to the reestablished second in-memory heap.

10 110. The carrier medium of claim 100,

 wherein the first process and second process are Java processes, and wherein the virtual machine is a Java virtual machine.

15

002050-31025560